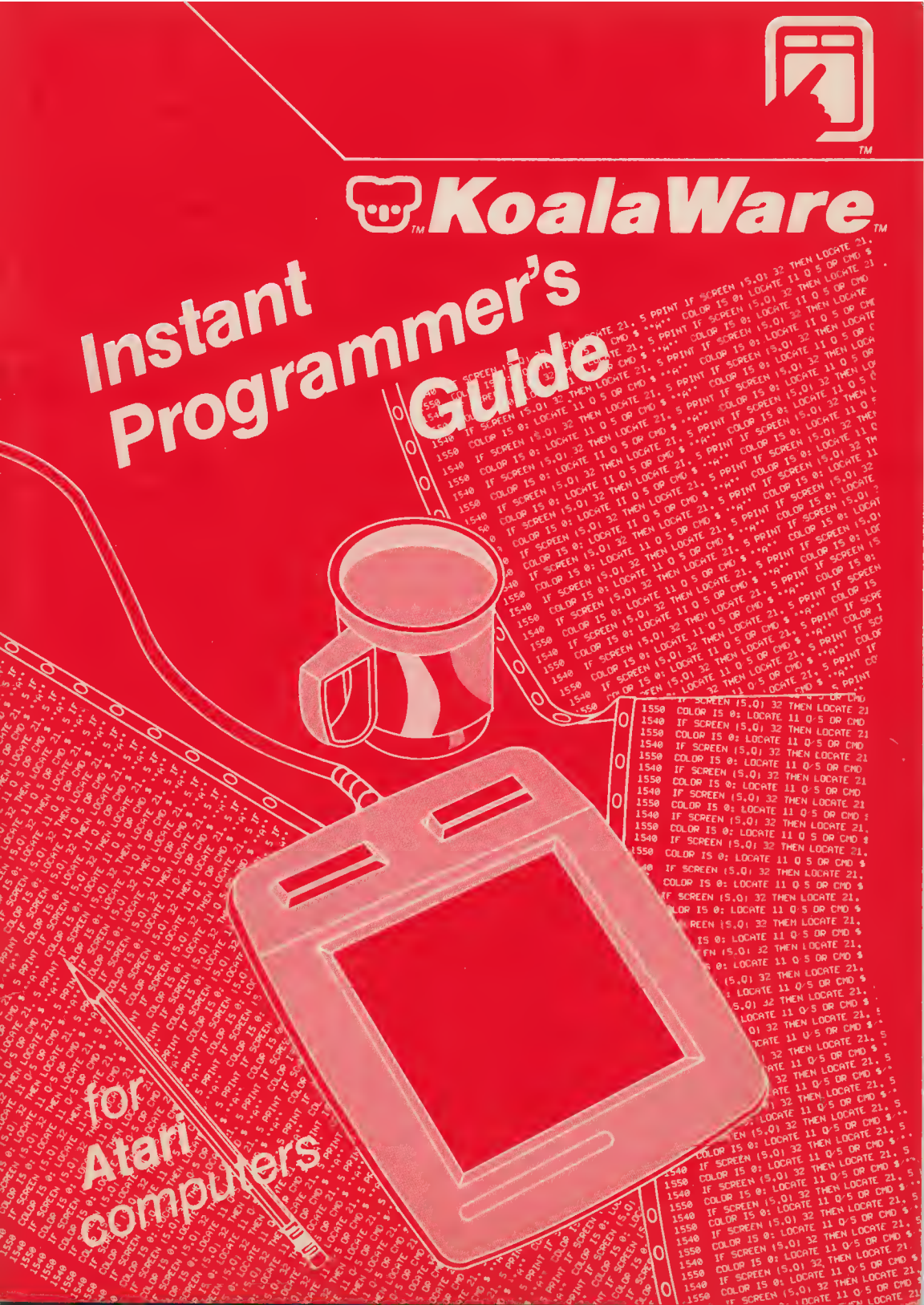
KoalaWare™

Instant Programmer's Guide



for Atari computers

Instant Programmer's Guide

for
Atari
computers

We make computing more personal.™

KOALAWARE INSTANT PROGRAMMER'S GUIDE

Atari Version

By Kerry S. Kurasaki and Sumiko Endo
Manual by Charley Chell

TABLE OF CONTENTS

INTRODUCTION

Congratulations! You are about to enter the world of free cursor movement using the most versatile input tool ever invented for microcomputers — the KoalaPad™ Touch Tablet.

The KoalaPad Touch Tablet is a state-of-the-art, highly sophisticated position sensing device. It converts finger or stylus pressure or movement into electronic signals for controlling computers. As such, it is particularly useful for drawing pictures and pointing to images on monitors.

With appropriate programming, a finger's movement can result in the drawing of a colored line on the display, the movement of a game piece, the composition of music, or the triggering of programmable function keys.

The Instant Programmer's Guide is a collection of tools for learning about the KoalaPad Touch Tablet. Many techniques for reading and using the tablet are covered. Short cuts and hints to successful, easy programming using the KoalaPad Touch Tablet are made available throughout this guide.

After reviewing these topics, hints, and examples, you will be programming using the KoalaPad Touch Tablet in no time.

Go To It!

SYSTEM REQUIREMENTS

- Atari Computer with at least 48K of RAM
- Atari 810 or 1050 disk drive
- Color television or video monitor
- Atari BASIC Cartridge
- KoalaPad Touch Tablet and Stylus
- Instant Programmer's Guide diskette

SYSTEM OPERATION

1. Make sure your computer, disk drive, KoalaPad Touch Tablet, and television monitor are correctly connected. The KoalaPad plugs into Joystick Port 1 of your Atari computer.
2. Turn on your television and disk drive, then insert the Instant Programmer's Guide diskette into your disk drive.
3. Making sure that the BASIC cartridge is inserted, turn on your computer and wait for the READY prompt from BASIC.
4. Type RUN "D:HELLO" return

KOALAPAD FUNDAMENTALS

Reading Raw Data

The KoalaPad Touch Tablet returns X and Y values corresponding to the placement of the finger or stylus on the pad. Tablet values range up to a maximum of 228 on each axis.

For technical reasons the tablet cannot quite generate a full range of values 0-228. Rather, the tablet minimum will typically be about 4 (or less) on each axis.

To correspond to graphics screens, the origin of the tablet, i.e., point (0,0) is located in the upper left corner.

The KoalaPad returns a position even if it is not pressed. In the "release" or "inactive" state the value will typically be less then (4,4) but **NOT** zero. You'll learn more about this later.

Getting the X and Y values from the tablet in BASIC is identical to what you do when reading paddles:

```
100 X = PADDLE(0)
110 Y = PADDLE(1)
```

Detecting Button Presses

The KoalaPad Touch Tablet is equipped with two buttons that may be read and used within a program.

The tablet buttons look like trigger buttons. Button presses may be read by either of two methods:

```
100 LEFT = PTRIG(0)
110 RIGHT = PTRIG(1)

100 LR = STICK(0)
110 IF LR=15 THEN 1000:REM      NEITHER BUTTON
120 IF LR=11 THEN 2000:REM      LEFT BUTTON PRESSED
130 IF LR=7 THEN 3000:REM       RIGHT BUTTON PRESSED
140 IF LR=3 THEN 4000:REM       BOTH BUTTONS PRESSED
```

When a button is in a pressed state, it returns a value of 0. When a button is in a released state, it returns a value of 1.

Drawing with the Tablet

The KoalaPad Touch Tablet is ideal for all types of plotting applications. Plotting points is simple. The main consideration is to limit the tablet values so that they don't exceed the screen and cause BASIC to abort the program.

The following example demonstrates how to plot points.

```
100X=PADDLE(0):Y=PADDLE(1):REM      READ TABLET
110 L=PTRIG(0):R=PTRIG(1):REM       READ BUTTONS
120 IF X159 THEN X=159:REM      LIMIT X AND Y VALUES
130 IF Y79 THEN Y=79
140 RETURN

200 GRAPHICS 7:COLOR 1:REM      SET UP FOR GRAPHICS
210 GOSUB 100:REM       GET X,Y
220 PLOT X,Y
230 IF L=1 AND R=1 THEN 210:REM      CONTINUE IF NO BUTTON
```

Lines can be drawn just as easily by making a few modifications to the example:

```
300 GRAPHICS 7:COLOR 1:REM      SET UP FOR GRAPHICS
310 GOSUB 100:REM      GET X,Y
320 PLOT X,Y:OLDX=X:OLDY=Y:REM      PLOT INITIAL POINT
330 GOSUB 100:REM      GET X,Y
340 IF OLDX-X>20 THEN 330:REM      THROW OUT BAD VALUES
350 IF OLDY-Y>20 THEN 330
360 DRAWTO X,Y:OLDX=X:OLDY=Y
370 IF L=1 AND R=1 THEN 330:REM      CONTINUE IF NO BUTTON
```

Insufficient pressure while drawing may cause the generation of erroneous readings. This is because the tablet is fluttering between the point and release states. Proper programming (use of the OLDX and OLDY variables) can filter out these bad readings.

The filtering technique demonstrated in the Guide works by comparing the current tablet reading to the last reading. If the new reading is sufficiently far from the old reading, then the new reading is assumed to be in error. Since erroneous tablet readings are the result of light pressure, they will always be less than the desired value. Therefore the check is only for values less than the old value.

The drawback of this simple filter is that it limits the speed at which the stylus or finger may be moved. The filter can be improved by making the tolerance value (20) a function of the distance between the last two good tablet values. This filter then self-adjusts to the stylus speed.

Scaling Data

The previous examples on drawing use only a portion of the tablet surface. This is because lines 120 and 130 were used to clip the tablet values.

An alternative to clipping is to scale the values instead. By scaling, the entire tablet surface may be used no matter which graphics mode is in effect.

The code sample shown below performs scaling appropriate to the Atari computer graphics mode 7:

```
100 X=PADDLE(0):Y=PADDLE(1):REM     READ TABLET
110 L=PTRIG(0):R=PTRIG(1):REM     READ BUTTONS
120 X=X*SX:Y=Y*SY:REM     SCALE
130 RETURN
200 SX=179/228:REM     X SCALE FACTOR
210 SY=79/228:REM     Y SCALE FACTOR
300 GRAPHICS 7:COLOR 1:REM     SET UP FOR GRAPHICS
310 GOSUB 100:REM     GET X,Y
320 PLOT X,Y
330 IF L=1 AND R=1 THEN 310:REM     CONTINUE IF NO BUTTON
```

Table scaling can also be used to generate points below the release value. Subtract the release value from the raw tablet value and then scale by the adjusted factor. The modifications below illustrate this.

```
120 X=(X-4)*SX:Y=(Y-4)*SY
200 SX=179/(228-4):SY=79/(228-4)
```

FUN WITH THE KOALAPAD

Sound

The use of the KoalaPad isn't limited to drawing. The X and Y values of the Touch Tablet work quite nicely to control the sound capabilities of the Atari computer. The example shown below uses the X and Y values to control the sound volume and pitch respectively.

```
200 X = PADDLE(0)/23 + 5:Y = PADDLE(1):REM    GET TABLET VALUES
210 SOUND 0,Y,10,X
220 IF PTRIG(0) = 1 AND PTRIG(1) = 1 THEN 200
230 SOUND 0,0,0,0:REM    SHUT OFF SOUND
```

This example makes use of only one of the four sound channels on the Atari and does nothing with the distortion. It doesn't take too much imagination however, to see that a clever programmer can use the tablet to control the Atari in a much more complicated fashion -- or even to implement a small keyboard.

Assembly Language Support

This section and the next introduce a machine language package tablet interface and graphics package. The package provides several routines, listed in the appendix.

The BASIC programs of the Atari Programmer's Guide are the best reference for how to load and use the machine language package. The AUTORUN . SYS file on the Instant Programmer's diskette must be executed to reserve memory for the package.

Rubber-Band Lines and Boxes

One of the main features of the Atari Programmer's Guide machine language package is its ability to draw what has been accurately described as "Rubber-Band Lines and Boxes." The way to describe them is by demonstration, so if you haven't run the FUN WITH THE KOALAPAD section of the program, do so now.

The samples below demonstrate one technique for producing rubber lines and boxes. The examples make extensive use of the machine language routines.

```
300 MODE=0:REM    RUBBER LINE
310 Z=USR(CURMOD):REM    GET CURSOR POSITION
320 AY=INT(Z/256):AX=Z-AY*256:REM    POS IS ANCHOR PT
330 Z=USR(RUBLBX,AX,AY,MODE):REM    RUBBER ROUTINE
340 Y=INT(Z/256):X=Z-Y*256:REM    RETURNS CURSOR POS
350 PLOT AX,AY:DRAWTO X,Y:REM    DRAW LINE

400 MODE=0:COLOR=1:REM    RUBBER BOX
410 Z=USR(CURMOD):REM    GET CURSOR POSITION
420 AY=INT(Z/256):AX=Z-AY*256:REM    POS IS ANCHOR PT
430 Z=USR(RUBLBX,AX,AY,MODE):REM    RUBBER ROUTINE
440 Y=INT(Z/256):X=Z-Y*256:REM    RETURNS CURSOR POS
450 Z=USR(DRWBOX,AX,AY,X,Y,COLOR):REM    DRAW BOX
```

TEXT CURSOR

Of course, the Touch Tablet is ideal for pointing to text. As demonstrated in the program, the tablet can move a text cursor which can be used by the software to mark characters, words, lines, paragraphs, etc. This capability could be the foundation of a good word processor.

TIC TAC TOE

A two-person tic-tac-toe game is provided to illustrate another use of the tablet. Think of the tablet as a 3 by 3 array of buttons that correspond to the tic-tac-toe board.

The tablet makes an excellent soft keyboard. The "keys" of the tablet keyboard are assigned by the software and thus may be any size and shape. Try programming the tablet to work as a numeric keypad.

COMPATIBILITY

The KoalaPad Touch Tablet can run much, but not all, of your existing software that uses paddles. The best way to find out how it works with any particular package is to try it.

Incompatibility mainly arises in two player games. Generally they need two separate inputs, one under the control of each player. The tablet always sends both paddle inputs at once and cannot be controlled by two people.

TOUCH TABLET OVERLAYS

In the technical portion of this Guide you learned how to program using the touch tablet. Here, we'll give you suggestions about how to enhance your applications of the tablet using overlays.

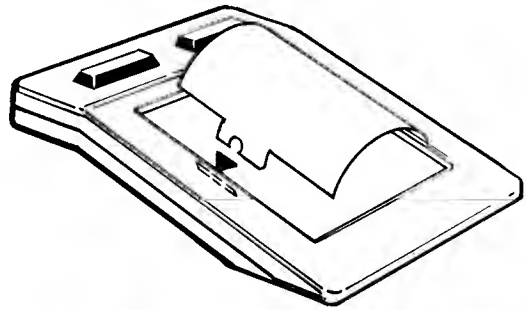
An overlay is a durable cover which fits on the active surface area of the tablet. It could have a grid or design printed on it which would help the user point to certain areas on the video monitor.

An overlay could easily turn the surface of the tablet into 49 "keys" for programmable functions. Division of the tablet into more than a 7 by 7 array is not advised because the buttons will be too small to be pressed easily. Also, the perimeter of the surface area within an eighth of an inch from the edge might not be suitable for use as a key. This is due to the manufacture of the tablet.

The ways an overlay can be used are as varied as the software. We've used it as a piano keyboard in a music program, and as a design tool for pre-programmed figures in a cartoon product. Other applications might be icon selection, soft keyboards, templates, and grids for design work.

Creating an Overlay

1. Create a drawing on paper which fits the size of the clear, blank overlay included with this package.
2. You can either make the lines in the drawing very bold yourself, or take it to a typesetter or printer to have a photostat made of it.
3. Cut out the drawing along the border to match the included overlay.
4. Place the paper drawing on the tablet surface and the clear overlay on top of it as shown here.



Technical Specifications

KoalaPad Touch Tablet

Size	6″ X 8″ X 1″
Active Surface Area	4.25″ X 4.25″
Weight	Approx. 1 lb.
Resolution	Typically 228 points per X-Y axis for the Atari
Power Requirements	+5v Approx. 20 ma.
Template Overlay Locations	Designed Into Housing
Operating Temperature and Humidity	0 to 50° C. 95% RH non-condensing

APPENDIX

This appendix describes the function interfaces to the Atari Programmer's Guide machine language tablet and graphics package.

The table below summarizes the functions. Following that are details of the functional interface of each.

Routine	Address	Description
INIT	8192	Package initialization, call this one first
FINISH	8195	Cleanup routine, call this one last
XGET	8198	Returns current tablet X value
YGET	8201	Returns current tablet Y value
DRWPT	8204	Plots a point
DRWVEC	8207	Plots a line
DRWBOX	8210	Plots a box
CURMOD	8213	Enters cursor mode
RUBLBX	8216	Rubber Line/Box routine

Loading Procedures

INIT

X = USR(INIT)

Takes no other arguments, returns no value. Used to link the filter code into the interrupt system. If not called, invalid points will be returned.

FINISH

X = USR(FINISH)

Takes no other arguments, returns no value. Used to unlink filter code from the interrupt system. This routine must be called at the completion of the program or after a program break for proper exit to the DOS operating system.

GETX

X = USR(GETX)

Takes no other arguments, returns the current tablet X coordinate. This routine performs exactly as PADDLE(0) and is supplied because it is used internally.

GETY

X = USR(GETY)

Takes no other arguments, returns the current tablet Y coordinate. This routine performs exactly as PADDLE(1) and is supplied because it is used internally.

DRWPT

Z = USR(DRWPT,X,Y,COLOR)

Plots a point at (X, Y). Takes three arguments. The first two are coordinates and the third is the color. COLOR may be 0, 1, 2, 3, or 255.

Color 0 is background. Color 255 signifies a special mode used in rubber-band lines and boxes where an XOR of the color is done. I.e., 0-3, 1-2, 2-1, 3-0.

DRWVEC

Z = USR(DRWVEC,X1,Y1,X2,Y2,COLOR)

Draws a line from (X1, Y1) to (X2, Y2). Color is treated as in DRWPT.

DRWBOX

Z = USR(DRWBOX,X1,Y1,X2,Y2,COLOR)

Draws a box from (X1, Y1) to (X2, Y2). Color is treated as in DRWPT.

CURMOD

POS = USR(CURMOD)

Takes no other arguments. Returns current cursor screen position. High byte contains Y position. Low byte contains X position.

RUBLBX

Z = USR(RUBLBX,AX,AY,MODE)

Rubber line and box routine. Called with an anchor point, (AX,AY), returns on a button press. While the routine is executing, it continuously draws a line or box from the anchor point to the current tablet position. MODE = 0 for line, MODE = 1 for box.

Notes

Disclaimer of Warranties and Liability

Even though Koala Technologies has tested the product described in this manual and reviewed its contents, and has made every attempt to verify the accuracy of this manual and its accompanying software, **NEITHER KOALA TECHNOLOGIES NOR ITS SUPPLIERS MAKE ANY WARRANTY, EXPRESSED OR IMPLIED, WITH RESPECT TO THIS MANUAL OR TO THE SOFTWARE DESCRIBED IN THIS MANUAL, AS TO THEIR QUALITY, PERFORMANCE, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE.** As a result, this product and manual are sold "as is", and you the purchaser are assuming the entire risk as to their quality and performance. In no event will Koala Technologies or its suppliers be liable for direct, indirect, incidental, or consequential damages resulting from any defect in the product or manual, even if they have been advised of the possibility of such damages. In particular, they shall have no liability for any programs developed with this product, including the costs of recovering or reproducing these programs or data. Some states do no allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you.

Part No. 108-00248-004

Printed in USA Rev. 10/83